第 37 卷第 1 期
录用定稿
计算机应用研究
Application Research of Computers
ChinaXiv合作期刊
Vol. 37 No. 1
Accepted Paper

融合协同过滤和 XGBoost 的推荐算法 *

崔　岩，祁　伟，庞海龙，赵　辉

(长春工业大学 计算机科学与工程学院，长春 130012)

摘　要：协同过滤是信息过滤和推荐系统中应用最广泛的技术，但是在数据处理中存在数据稀疏问题，影响推荐算法的准确性。提出融合协同过滤和 XGBoost 的推荐算法，根据用户对项目的评价以及项目本身所具备的自身特点，挖掘项目和用户的潜在关系，提高算法的推荐准确性。采用百度深度学习框架 PaddlePaddle 在 Book-Crossings 数据集上进行实验，实验结果表明，提出的算法和文献中两种算法相比，准确性有显著提升。

关键词：XGBoost；协同过滤；准确性；推荐系统

中图分类号：TP301.6　　doi: 10.19734/j.issn.1001-3695.2018.06.0463

Extreme gradient boosting recommendation algorithm with collaborative filtering

Cui Yan, Qi Wei, Pang Hailong, Zhao Hui

(*College of Computer Science & Engineering*, *Changchun University of Technology*, *Changchun* 130012, *China*)

Abstract: Collaborative filtering plays an important role in recommendation system and is the most successful and widely used technology in information filtering and information system. However, collaborative filtering has a sparse problem in data processing, which affects the accuracy of the proposed algorithm. This paper proposed a recommendation algorithm combining collaborative filtering and XGBoost to explore the potential relationship between the project and the user based on the user's evaluation of the project and its own characteristics. It improved the recommendation accuracy of the algorithm. The results of experiments on the book-crossings data set using the baidu deep learning framework paddlepaddlepaddles show that, Compared with the two algorithms in the literature, the accuracy of the proposed algorithm is significantly improved.

Key words: XGBoost; collaborative filtering; accuracy; recommendation system

0　引言

随着近些年来网络环境的迅速发展，网络信息的覆盖正在遍及本文所需要的各个方面。人们在线获取的数据也越来越丰富，但却导致了数据量急速增长，根据数据统计结果表示，在每分钟时间内，Facebook 的活跃用户会在网络上分享大约 68.4 万比特的信息，Twitter 用户则会发出超过 10 万条，世界上 90%的数据产生在 2010—2012 年，到 2020 年，全球信息总量将会是 2011 年的 22 倍，达到 35.2 ZB[1]。但其中有很多属于无关冗余数据，这导致了“信息超载[2]（information overload）”问题，网络世界被信息所包围，正在从 IT(information technology)走向 DT（data technology）时代[3]，因此，推荐系统[4]应运而生，成为帮助用户获取有效信息的必要工具，作为一种解决信息量超载的过滤技术，起到了重要的作用。

在推荐系统发展中，协同过滤[5]仍占据着主导位置。协同过滤能够过滤机器难以自动进行内容分析的信息，有效地利用其他相似用户的反馈信息、提高个性化学习的速度。但实际应用中会出现数据稀疏性[9-10]问题、数据增加情况下的扩展性[11]问题、冷启动[12]问题。数据稀疏性是在用户对项目评价很少时，基于用户的评价所得到的用户间的相似性可能达不到预期效果，最后会影响推荐的准确性，扩展性问题是：在数据量很大时或新加入大量信息后，不能及时找出相似度很高的用户项目关系，从而无法实现实时推荐。冷启动问题是当新用户与新项目加入推荐系统时，因为没有相应的历史数据，无法进行相似度计算，从而无法产生有效推荐。

针对上述问题，已有许多相关研究。文献[13]提出两阶段联合聚类评分方法，聚类后的矩阵维度远远小于原始矩阵维度，减少计算量的同时又提高了精准度，但算法须依靠评分的权重，预测得到的评分可靠度不高。文献[14]提出了奇异值分解（SVD）解决维度问题，解决了数据量增加情况下的扩展问题，但在数据量很大时十分耗费时间，不利于使用。文献[15]提出了融合社交信息的矩阵分解推荐方法，通过社交信息能够更加准确的挖

收稿日期：2018-06-21；**修回日期：**2018-08-07　**基金项目：**国家自然科学基金资助项目（61472049）；吉林省教育厅“十二五”科学技术研究项目（2014132）

作者简介：崔岩（1993-），男，硕士研究生，主要研究方向为推荐系统、智能计算；祁伟（1970-），女（通信作者），讲师，硕士，主要研究方向为智能计算，搜索引擎（qiwei@ccut.edu.cn）；庞海龙（1987-），男，吉林长春人，硕士研究生，主要研究方向为推荐系统、智能计算；赵辉（1972-），女，教授，博士，主要研究方向为智能计算、搜索引擎．

掘兴趣爱好，并通过好友关系提高了推荐的精准度，在文献[16]中提出了 XGBoost[17]算法在电子商务商品推荐中的应用,不足在于未能联系用户以往的行为特征，推荐存在缺陷。

本文为了解决数据的稀疏性提出了一种融合协同过滤和 XGBoost 的推荐算法（eXtreme Gradient Boosting recommendation algorithm with collaborative filtering, XGBCF）通过协同过滤构建用户间的相似矩阵和项目间的相似矩阵计算相似度，根据最近邻关系组成相似集合对，生成训练集，再利用 XGBoost 对训练集进行拟合，得出最佳权值和学习率，利用目标函数进行分类,推荐给用户，提高了推荐的准确性。

1 相关工作

1.1 协同过滤算法

协同过滤（collaborative filtering recommendation）在信息过滤和信息系统中是一种常用的技术，是集体智慧的一种典型体现，其原理是对用户的历史行为进行分析和挖掘,发现用户的偏好，基于不同的兴趣偏好对用户进行群组划分。最主要的功能为预测和推荐，协同过滤主要分为两类：基于用户的协同过滤算法（user based collaborative filtering）和基于项目的协同过滤（item-based collaborative filtering），基于用户的协同过滤算法分析了用户的历史行为，找出用户感兴趣的内容或项目，并根据不同用户对相同项目的打分，分析他们的偏好程度，计算相似度，从而在相似用户间进行推荐。基于项目的协同过滤原理和基于用户的协同过滤相似，只是在计算相似度的时候关注的是项目本身，也就是说根据用户对项目的偏好找到相似的项目，其步骤为先计算已评价项目和待预测项目的相似度，并以计算出的相似度作为权重系数，通过权重控制已评价项目的分数，得到待预测项目的预测值。与已评价的进行比较，如果预测值和已评价的较相近便进行组合推荐。

关于相似度的计算，现有的几种基本方法都是基于向量下的相似度计算，即计算向量之间的距离，距离越近，他们的相似度越大。推荐的情况下，在用户物品偏好的二维矩阵中，本文可以通过将用户对所有物品的偏好作为矢量来计算用户之间的相似度，或者将所有用户对某个项目的偏好作为矢量来计算项目之间的相似度。其常用的相似度包括：皮尔森相关系数[18]（Pearson correlation coefficient，PCC），余弦相似性[19]等。

余弦相似度（cosine similarity）计算公式如下：

$$sim(u,v)=\frac{\sum_{i\in I}R_{u,i}\bullet R_{v,i}}{\sqrt{\sum_{i\in I}R_{u,i}^{2}}\bullet\sqrt{\sum_{i\in I}R_{v,i}^{2}}} \tag{1}$$

其中：所求的是用户 u 和 v 的相似度，I 为两个用户都评过分的项目，$r_{u,i}$ 表示用户 u 对项目 i 的评分，$r_{v,i}$ 表示的是用户 v 对项目 i 的评分。

皮尔森相似系数(Pearson correlation coefficient）定义为

$$sim(u,v)=\frac{\sum_{u\in U}(R_{u,i}-\overline{R_i})(R_{u,v}-\overline{R_v})}{\sqrt{\sum_{u\in U}(R_{u,i}-\overline{R_i})^2}\sqrt{\sum_{u\in U}(R_{u,v}-\overline{R_v})^2}} \tag{2}$$

其中：U 表示的是所有评过分数的用户集合，$\overline{R_i}$ 表示的是对项目 i 的平均打分，$\overline{R_j}$ 标志的是对项目 j 的平均打分。皮尔森相关系数用于计算两个用户联系的紧密程度，取值在[-1,1]。

协同过滤预测评分公式定义为

$$R_{v,i}=\overline{r_v}+\frac{\sum_{u\in U}(r_{u,i}-\overline{r_u})*sim(u,v)}{\sum_{u\in U}|sim(u,v)|} \tag{3}$$

其中：$R_{v,i}$ 表示用户 v 对为评分项目 i 的预测值，V 表示的是最近邻中拥有待评分项 i 的用户集合。

Top-*N* 算法是一种按照一定规则对数据进行排序的算法，在本文中将本文提出的算法（XGBCF）的结果通过式（3）计算未对项目评分用户的预测评分值，然后把预测值按照降序排列，产生推荐列表，推荐给用户。

1.2 XGBoost 算法

XGBoost 全称为 extreme gradient Boosting,是华盛顿大学陈天奇博士 2014 年在 GBDT 算法基础上对 Boosting 算法提出的的一种改进，内部决策树使用的是回归树，具有速度快，效果好，能够处理大规模数据，自定义损失函数等一系列特点。它采用了弱分类器的逐次迭代计算，提高分类的精确性。算法相关原理如下：

树的复杂性定义为

$$f_t(x)=w_{q(x)},w\in R^T,q\in R^d\rightarrow\{1,2,...,T\} \tag{4}$$

其中：q 表示的是树的结构，w 表示叶子权重部分。

改进的复杂度函数为

$$\Omega(f_t)=\gamma^T+\frac{1}{2}\lambda\sum_{j=1}^{T}w_j^2 \tag{5}$$

其中：T 为叶子节点个数，w_j^2 表示的是每个叶子节点上面输出分数的 L_2 模平方，γ和λ 在最终的模型公式中控制这部分的比重.这样就可以衡量模型的复杂性，从而有效控制过拟合。

改写的目标函数：

$$\begin{aligned}Obj^{(t)}&=\sum_{i=1}^{n}[l(y_i,\hat{y}_i^{(t-1)})+g_i f_t(x_i)\\&+\frac{1}{2}h_i f_t^2(x_i)]+\Omega(f_t)+cons\tan t\end{aligned} \tag{6}$$

其中：*l* 就是损失函数，Ω（f_t）为正则化项，包含 L_1、L_2 正则化，constant 为常数项，通过式子可以看出，最终的目标函数依赖于每个数据点的在误差函数上的一阶和二阶导数。

利用式（6）对目标函数再次改写，其中 l 被定义为每个叶子节点上面样本集合 $I_j=\{i\,|\,q(x_i)=j\}$ 。

$$\begin{aligned}Obj^{(t)}&\cong\sum_{i=1}^{n}[g_i f_t(x_i)+\frac{1}{2}h_i f_t^2(x_i)]+\Omega(f_t)\\&=\sum_{j=1}^{T}[(\sum_{i\in I_j}g_i)w_j+\frac{1}{2}(\sum_{i\in I_j}h_i+\lambda)w_j^2]+\lambda^T\end{aligned}$$

定义 $G_j=\sum_{i\in I_j}g_i$ ， $H_j=\sum_{i\in I_j}h_i$

化简为　$$Obj^{(t)}=\sum_{j=1}^{T}[G_j w_j+\frac{1}{2}(H_j+\lambda)w_j^2]+\gamma^T \tag{7}$$

对 w_j 进行求导，令导数为 0 可以得到：

$$w_j^*=-\frac{G_j}{H_j+\lambda}$$

把 w_j 最优解代入得到目标函数：

$$Obj=-\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda}+\gamma^T \tag{8}$$

其中：T 为叶子节点的个数，λ和γ 为比重系数，防止过拟合的产生。

2　融合协同过滤和 XGBoost 的推荐算法构建

2.1　算法思想

因每个用户对项目的评价有限，在构建用户行为矩阵时会存在大量空值，为了解决此问题，本算法根据用户的以往行为，挖掘用户间关系和用户项目间关系，通过已评分项目构建用户对和用户项目对，并分别计算相似度，构建相似矩阵，计算每个用户对其他用户及项目的相似性，对计算结果进行排序，组成最近邻集合，通过协同过滤预测评分公式，得到预测评分，再利用 XGBoost 算法进行分类，计算出分类错误率，通过权值和学习率的更新，找出分类错误的样本，并重置权重，达到提高算法精准度的效果，最后利用 top-*N* 生成推荐列表展示给用户。

2.2　算法相关定义

定义 1　用户集合U,共有n个用户，下标表示用户的索引。

$$U=\{u_1,u_2,....u_n\}$$

定义 2　项目集合 I，共有 m 个项目，下标表示项目的索引。

$$I=\{I_1,I_2,.....I_m\}$$

定义 3　表 1 中为项目和特征的交叉，$if_{m,n}$ 表示的是第 m 个项目的第 n 个特征.

表 1　项目特征矩阵 ifeature

Table1 Item feature marix ifeature

项目	f_1	f_2	f_3	...	f_m
i_1	$if_{1,1}$	$if_{1,2}$	$if_{1,3}$	...	$if_{1,n}$
i_2	$if_{2,1}$	$if_{2,2}$	$if_{2,3}$	...	$if_{2,n}$
...	...	...	...	...	...
i_n	$if_{n,1}$	$if_{n,2}$	$if_{n,3}$	...	$if_{m,n}$

定义 4 用户特征矩阵 ***ufeature*** 表格中 U_{pq} 为用户和特征的交叉，表示的是第 p 个用户的第 q 个特征。

表 1　用户特征矩阵 ufeature

Table2 User feature marix ufeature

用户	f_1	f_2	f_3	...	f_p
u_1	$uf_{1,1}$	$uf_{1,2}$	$uf_{1,3}$	...	$uf_{1,p}$
u_2	$uf_{2,1}$	$uf_{2,2}$	$uf_{2,3}$	...	$uf_{2,p}$
...	...	...	...	...	...
u_q	$uf_{q,1}$	$uf_{q,2}$	$uf_{q,3}$	...	$Uf_{p,q}$

定义 5 用户对项目评分矩阵 R,其中 $R_{p,m}$ 表示的是用户 p 对项目 m 的评分。

表 3　项目评分矩阵

Table3 Item scoring marix ifeature

用户	i_1	i_2	i_3	...	i_m
u_1	$R_{1,1}$	$R_{1,2}$	$R_{1,3}$	...	$R_{1,m}$
u_2	$R_{2,1}$	$R_{2,2}$	$R_{2,3}$	...	$R_{2,m}$
...	...	...	...	...	...
u_p	$R_{p,1}$	$R_{p,2}$	$R_{p,3}$	...	$R_{p,m}$

2.3　算法描述

本算法包括生成训练样本集，XGBoost 精准分类处理和生成推荐三个步骤：

1)生成训练样本集

输入：经过数据清洗的用户特征矩阵 ufeature，经过数据清洗的项目特征矩阵 ifeature，用户对项目的评分矩阵 R，用户集合 U，项目集合 I。

输出：训练样本集。

处理流程：

a)在用户—项目的评分矩阵中找出已经评分的项目集合 I 和已经对项目评过分数的用户集合 U。

b)把上述的 I 和 U 分别拆分成两个为一组的用户对集合 $upair=\{<u_m,u_n>|u_m,u_n\in U\}$ 和项目对集合 $ipair=\{<i_a,i_b>|i_a,i_b\in I\}$。

c)找出 *upair* 和 *ipair* 拆分的数据对在 *ifeature* 和 *ufeature* 所对应的位置，然后用式（1）计算出用户对相似度 $sim(u_m,u_n)$，利用式（2）计算项目对相似度 $sim(i_m,i_n)$。

d)循环前三个步骤，得到所有用户和所有项目对应相似性，分别构建用户相似度矩 $usim(m,m)$ 和项目相似度矩阵 $isim(n,n)$。

e)计算每个用户在用户相似度矩阵的相似性，对计算结果进行排序，将相似度最高的 n 个用户组合成用户最近邻集合 $N_u=\{(u_1,u_2....u_n)|u\in usim\}$，同样方式组合项目最近邻集合 $N_i=\{(i_1,i_2...i_n)|i\in isim\}$。

f)遍历用户对项目的评分矩阵 R，选出项目 i 对应的最近邻集合 N_i。根据式（3）计算出用户 p 对项目 n 的预测评分 $p_{p,n}^i$，把预测值和原始用户-项目矩阵中的差值记作 x_1^i，同理选出用户 u 对应的最近邻集合 N_u，利用式（3）计算出用户 p 对项目 n 的预测评分 $P_{p,n}^u$，把预测值和原始用户-项目矩阵中的差值记作 X_2^u，最后把上面得到的差值和 $R_{u,i}$ 组合成新的数据集 $data=\{\{x_1^i,x_2^u,R_{u,i}\}|i\in I,u\in U,R_{u,i}\in R\}$

2)XGBoost 分类

输入：训练集样本。

输出：分类结果。

处理流程：

a)将所有的训练样本赋予相同的权重 w。

b)采用 XGBoost 算法进行分类，迭代 m 次，并用下面的公式计算分类错误率:

$$err_m = \frac{\sum \omega_i I(y_i \neq G_m x_i)}{\sum \omega_i} \qquad (9)$$

其中：ω_i 表示第 i 个样本的权重，G_m 代表的是第 m 个分类器。

c)更新学习率：

$$\alpha_m = \frac{1}{2}\log((1 - err_m) / err_m) \qquad (10)$$

这样做的目的是通过剪枝防止过拟合的产生，同时降低了生成树的规模，保证迭代样本空间，在之后的学习率计算中可以通过迭代更新学习率。

d)反复进行步骤 b)c)，得到相应的错误率，然后重置第 n 个样本的权重为 $\omega_0 = \omega_i * e^{\alpha_m * I(y_i \neq G_m x_i)}$

e)在经过多次迭代后，通过更新学习率和权重系数找出分类错误的样本，并进行权重重置，这样便可以得到更加精确地分类。

3)生成推荐列表

输入：数据集 data，用户 u 在 data 中的数据。

输出：用户 u 的推荐列表。

a)构建回归树，将数据集以标签的形式作为根节点输入。

b)根据式（8）所示的目标函数建立 XGBoost 模型，因为 XGBoost 有可以自定义损失函数的特点，在这里将损失函数定义为 logistic 损失函数。

$$l(y_i, \hat{y}_i) = y_i \ln(1 + e^{-\hat{y}_i}) + (1 - y_i)\ln(1 + e^{\hat{y}_i}) \qquad (11)$$

其中 y_i 表示实际值，$\hat{y}_i$ 标志的是预测值。

c)对损失函数求二阶偏导数，利用式（10）更新学习率，调整权重参数，代入到式(8)中，得到 XGBCF 模型。

d)利用 XGBCF 模型对用户 u 未评分的项目进行预测评分，并利用 top-N 产生推荐列表反馈给用户使用。

3　算法验证

3.1　实验数据

本文采用的数据集是由 Cai-Nicolas Ziegler 提供的 book-crossing[20]数据集，数据集由三部分组成，分别为用户信息数据集，图书信息数据集和用户评分数据集，其中用户数据集包括用户 id，人口统计信息（位置，年龄），如表 4 所示，图书数据集包含由各自的 ISBN 进行标识。无效信息已从数据集中删除。此外，还提供了一些基于内容的信息（‘书名’‘作者’‘出版日期’‘出版商’），如表 5 所示，评分数据集包含了 1,149,780 条评分信息，包含了 278,858 位用户对 271,379 本图书的评分，评级（Book-Rating）是明确的，区间从 1-10（更高的值表示更高的评价）也就是有更高的兴趣。实验中将数据集的 75%作为训练集，25%作为测试集。

表 4　用户信息特征示例

Table 4　Example of user information characteristics

用户 id	年龄	位置
1	18	Stock california
2	51	Black mountain

表 5　图书信息特征示例

Table 5　Example of book information characteristics

书名	作者	出版年份	出版商
Classical mythology	Mark p.o.morford	2002	Oxford university
Decision in Normandy	Richard bruce wright	2001	HarperFlamingo canada
More Cunning T	Amy Tan	2005	Berkley publish group

3.2　度量标准

本文实验采用平均绝对误差（mean absolute error ,MAE）度量分类准确性,MAE 在评分区间上做了归一化处理，定义如下：计算结果越小，预测越精准，推荐效果越好。

$$MAE = \frac{1}{|E^p|}\sum_{(u,a)\in E^p} |r_{ua} - r'_{ua}| \quad (12)$$

其中：r_{ua} 表示用户 u 对商品 a 的真是评分，r'_{ua} 表示的是用户 u 对商品 a 的预测评分，E^p 表示的是测试集。

3.3　实验结果与分析

因实验中不同参数会影响算法实验效果，所以先通过拟合数据集来确定一系列最优参数，再进行算法的比较，上文已经提及，本文的用户间相似度由皮尔逊相关系数计算，项目间相似度由修正的余弦函数相似性计算。

1)学习率计算

学习率控制算法损失梯度调整权值的速度，过小的学习率会导致收敛过慢，而过大又会导致代价函数动荡，在这里让学习率在 0.1~1 变化，通过实验图像（1）可以看出，在学习率 α =0.3 时，平均绝对误差（MAE）达到全局最小值，且不再随着学习率的增长而变化。

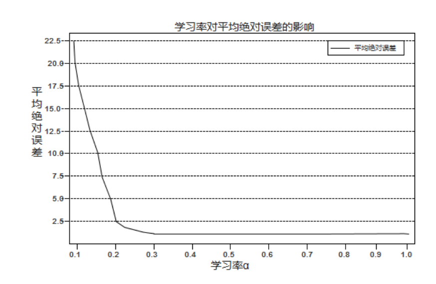


图 1　学习率对平均绝对误差的影响

Fig.1　Influence of learning rate on mean absolute error

2)最近邻数

最近邻数是影响推荐精准度的关键因素，本文已求出最佳学习率，在这里固定学习率 α =0.3，让最近邻数在 10-200 之间变化，通过实验求出最佳最近邻数，根据图 2 所示，随着用户最近邻数的增加，本文提出的融合算法的平均绝对误差逐渐减小，推荐精度上升，在最近邻数 k=125 时达到全局最小值并不再变化，如图 3 所示，最近邻数 k 在 10~75 区间变化时，XGBCF 的平均绝对误差震荡，随后增大 k 值，平均绝对误差（MAE）持续减小，推荐精度上升，项目最近邻数 k=125 时，XGBCF 的 MAE 达到最小，且不再发生变化，推荐精准率保持不变。

因此，针对于本文算法（XGBCF），分别设置用户最近邻数为 125，项目最近邻数为 125，学习率 α =0.3，进行实验。

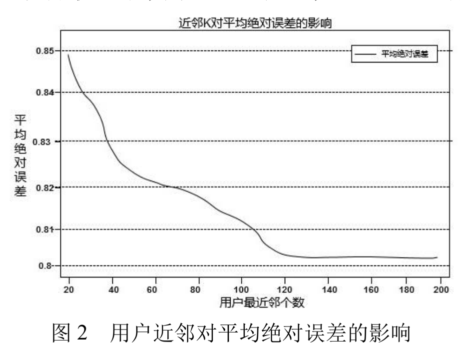


图 2　用户近邻对平均绝对误差的影响

Fig.2　Influence of user neighbour on mean absolute error

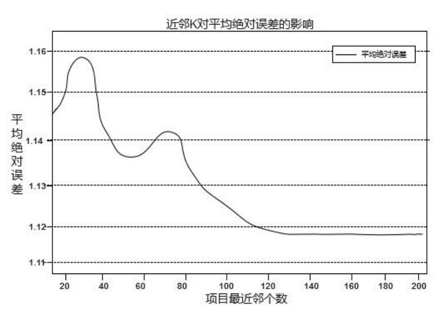


图 3　项目近邻对平均绝对误差的影响

Fig.3　Influence of item neighbour on mean absolute error

3)算法分析及对比

本文提出的算法（XGBCF）因加入了决策树，在构建算法时便对用户以及项目进行了初步分类，减少了后续的分类过程，在拟合过程中能够求出最优参数，达到了更精确的分类，推荐准确性也随之提高，将本文算法与文献[13]和文献[14]算法进行对比，根据图像（4）可以看出，本文提出的融合协同过滤和 XGBoost 的推荐算法的平均绝对误差（MAE）小于文献中提到的算法，可知本文提出的推荐算法准确性较文献中两个算法算法有显著提高。

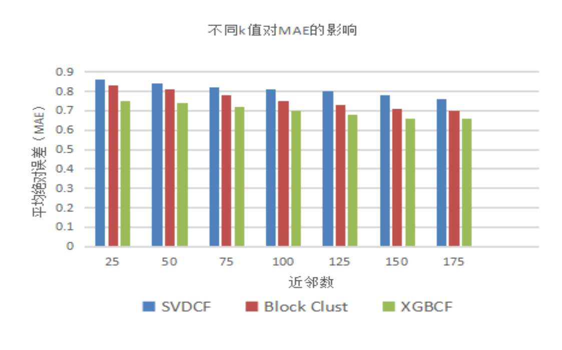


图 4　算法结果对比

Fig.4　Comparison of algorithm results

4　结束语

本文提出融合协同过滤和 XGBoost 的推荐算法，通过构建用户间的相似矩阵和项目间的相似矩阵计算相似度，根据最近邻关系组成相似集合对，生成训练集，再利用 XGBoost 对训练集进行拟合，得出最佳权值和学习率，利用目标函数进行分类。最后生成推荐列表推送给用户，本文算法克服了因数据稀疏导致的推荐准确性不足的现象，通过实验验证准确性较传统协同过滤有显著提高，下一步的工作将继续基于推荐系统的特征提取和安全问题进行进一步研究。

参考文献：

[1] Gantz J, Reinsel D. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. [EB/OL]. (2013-02) . https://www. emc. com/collateral/analyst-reports/idc-digital-universe-united-states. pdf.

[2] Meng Xiangwu, Hu Xun, Wang Licai, *et al.* Mobile recommender systems and their applications [J]. Journal of Software, 2013, 24 (1): 91-108.

[3] 连玉明. 人类社会从 IT 时代到 DT 时代 [J]. 商业文化, 2016 (11): 66-69. (Lian Yuming. Human society from IT era to DT era [J]. business culture, 2016 (11): 66-69.)

[4] Milicevic A K, Nanopoulos A Ivanovic, M. Social tagging in recommender systems: a survey of the state-of-the-art and possible extensions [J]. Artificial Intelligence Review, 2010, 33 (3): 187-209.

[5] Patil V A, Ragha L. Comparing performance of collaborative filtering algorithms [C]// Proc of International Conference on Communication, Information & Computing Technology. 2012: 1-6.

[6] 王成, 朱志刚, 张玉侠, 等. 基于用户的协同过滤算法的推荐效率和个性化改进 [J]. 小型微型计算机系统, 2016, 37 (3): 428-432. (Wang Cheng, Zhu Zhigang, Zhang Yuxia, *et al.* Recommendation efficiency and personalized improvement of user-based collaborative filtering algorithm [J]. Journal of Chinese Computer Systems, 2016, 37 (3): 428-432.)

[7] 马宏伟, 张光卫, 李鹏. 协同过滤推荐算法综述 [J]. 小型微型计算机系统, 2009, 30 (7): 1282-1288. (Ma Hongwei, Zhang Guangwei, Li Peng. Collaborative filtering recommendation algorithm [J]. Journal of Chinese Computer Systems, 2009, 30 (07): 1282-1288.)

[8] 刘建国, 周涛, 郭强, 等. 个性化推荐系统评价方法综述 [J]. 复杂系统与复杂性科学, 2009, 6 (3): 1-10. (Liu Jianguo, Zhou Tao, Guo Qiang, *et al.* Review of individualized recommendation system evaluation methods [J]. Complex Systems and Complexity Sciences, 2009, 6 (3): 1-10.)

[9] Yang Diyi, Chen Tianqi, Zhang Weinan, *et al.* Localimplicit feedback mining for music recommendation [C]// Proc of the 6th ACM Conference on Recommender Systems. New New ork: ACM Press, 2012: 91-98.

[10] Oh K J, Lee W J, Lim C G, *et al.* Personalized newsrecommendation using classified keywords to capture user preference [C]// Proc of the 16th International Conference on Advanced Communication Technology. Piscataway, NJ: IEEE Press, 2014: 1283-1287.

[11] 李聪. 电子商务协同过滤可扩展性研究综述 [J]. 现代图书情报技术, 2010 (11): 37-44. (Li Cong. Summary of research on scalability of collaborative filtering in electronic commerce [J]. Modern Library and Information Technology, 2010 (11): 37-44.)

[12] 孙小华. 协同过滤系统的稀疏性与冷启动问题研究 [D]. 杭州: 浙江大学, 2005. (Sun Xiaohua. The sparsity and cold start problem of collaborative



filtering system [D]. Hangzhou: Zhejiang University, 2005.)

[13] 吴湖，王永吉，王哲，等. 两阶段联合聚类协同过滤算法 [J]. 软件学报, 2010, 21 (5): 1042-1054. (Wuhu, Wang Yongji, Wang Zhe, *et al.* Two-stage joint clustering collaborative filtering algorithm [J]. Journal of Software, 2010, 21 (5): 1042-1054.)

[14] Vozalis M G, Margaritis K G. Applying SVD on item-based filtering [C]// Proc of International Conference on Intelligent Systems Design and Applications. Washington DC: IEEE Computer Society, 2005: 464-469.

[15] 刘华锋，景丽萍，于剑. 融合社交信息的矩阵分解推荐方法研究综述 [J]. 软件学报, 2018, 29 (2): 340-362. (Liu Huafeng, Jing Liping, Yu Jian. A review of matrix factorization recommendation methods for integrating social information [J]. Journal of Software, 2018, 29 (2): 340-362.)

[16] 张昊，纪宏超，张红宇. XGBoost 算法在电子商务商品推荐中的应用 [J]. 物联网技术, 2017, 7 (2): 102-104. (Zhang Hao, Ji Hongchao, Zhang Hongyu. Application of XGBoost algorithm in e-business commodity recommendation [J]. Internet of Things Technologies, 2017, 7 (2): 102-104.)

[17] Chen Tianqi, Carlos Guesintr. XGBoost: a scalable tree boosting system [C]// Proc of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2016: 785-794.

[18] Herlocker J, Konstan J, Borchers A, *et al.* An algorithmic framework for performing collaborative filtering [C]// Proc of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM Press, 1999: 230-237.

[19] Saltong. The SMART retrieval system-experiments in automatic document processing [M]. Englewood Cliffs, New Jersey: Prentice Hall Inc, 1971.

[20] GroupLens [EB/OL]. http://www. grouplens. org/.